Pure Cutting-Plane Algorithms and their Convergence

Dinakar Gade
Department of Industrial and Manufacturing Systems Engineering
3010 Black Engineering
Ames, IA 50010
dgade@iastate.edu
Simge Küçükyavuz
Department of Integrated Systems Engineering
210 Baker Systems, 1971 Neil Avenue
The Ohio State University, Columbus OH 43210
kucukyavuz.2@osu.edu

July 9, 2013

Abstract

Cutting-plane methods solve a mixed-integer program (MIP) by iteratively adding a valid linear inequality that violates a fractional solution of a linear relaxation of the problem. This paper surveys cutting-plane algorithms for different subclasses of MIPs and addresses whether these algorithms converge to an optimal solution of the MIP in finitely many steps.
Keywords: Cutting-plane algorithm, convergence, integer programming.

1 Introduction

This survey paper addresses the solution of the mixed-integer program (MIP)

$$z = \min_{x \in X}\{c^\top x : X = \{Ax \geq b, x \in \mathbb{Z}_+^p \times \mathbb{R}_+^{n-p}\}\}, \tag{1}$$

by means of adding valid linear inequalities (also referred to as cutting planes or cuts) to its linear programming relaxation. The *linear programming* (LP) *relaxation* of (1) is given by,

$$z_L = \min_{x \in X_\ell}\{c^\top x : X_\ell = \{Ax \geq b, x \in \mathbb{R}_+^n\}\}. \tag{2}$$

Here $c \in \mathbb{Q}^n$, $A \in \mathbb{Q}^{m \times n}, b \in \mathbb{Q}^m$. A *valid inequality* for the MIP (1) is an inequality $\pi^\top x \geq \pi_0$, where $(\pi, \pi_0) \in \mathbb{R}^{n+1}$ such that $X \subseteq \{x : \pi^\top x \geq \pi_0\}$. We refer the reader to Marchand et al. [29], Cornuéjols [18] and other articles in the encyclopedia for an introduction to valid inequalities for structured and unstructured MIPs.

A fundamental result in mixed-integer programming (see for example, Theorems 6.2 and 6.3 Chapter I.4 in Nemhauser and Wolsey [30]) states that solving problem (1) is equivalent to solving the linear program:

$$z = \min\{c^\top x : x \in \text{clconv}(X)\},$$

where $\text{clconv}(X)$ denotes the closure of the convex hull of X. Moreover, it can be shown that $\text{clconv}(X)$ is a polyhedron. However, constructing a linear description of $\text{clconv}(X)$ *a priori* is a difficult task and instead, the cutting-plane approach iteratively constructs tighter linear approximations of $\text{clconv}(X)$. Given a family of valid inequalities $\Pi := \{(\pi, \pi_0) \in \mathbb{R}^{n+1} : X \subseteq \{x : \pi^\top x \geq \pi_0\}\}$, a pure *cutting-plane* method for solving MIPs attempts to find a valid inequality (π, π_0) for X that cuts off the fractional solution $\bar{x}$ of the LP relaxation, i.e., $\pi^\top \bar{x} < \pi_0$. The problem of finding a violated inequality is referred to as separation. A generic cutting-plane algorithm for a given family of valid inequalities Π for X is given in Algorithm 1.

The algorithm begins with using the linear relaxation X_ℓ of X as an initial approximation of $\text{clconv}(X)$. In an iteration k of the algorithm, a linear program corresponding to the kth approximation X_ℓ^k is solved and the solution vector x^k is obtained. If x^k satisfies the mixed-integer requirement, then, because x^k is an optimal solution to a relaxation of (1), and feasible in X, it is also optimal. If this is not the case, then the algorithm proceeds by choosing a valid inequality $\pi^\top x \geq \pi_0$ that is violated by the current solution with $\pi^\top x^k < \pi_0$ (assuming that it exists in the given family), and the subsequent relaxation X_ℓ^{k+1} is formed by adding the valid inequality to X_ℓ^k. These steps are repeated until a mixed-integer feasible solution is found or until no valid inequalities to separate x^k can be found.

Algorithm 1 A Generic Cutting-Plane Algorithm for MIP

1: *Initialization.* $k \leftarrow 0, X_\ell^k \leftarrow X_\ell$.
2: **while** $x^k := \arg\min_{x \in X_\ell^k} c^\top x \notin \mathbb{Z}_+^p \times \mathbb{R}_+^{n-p}$ **do**
3:   *Separation.* Find an inequality $(\pi, \pi_0) \in \Pi$ such that $\pi^\top x^k < \pi_0$. If none exists, goto 6
4:   $X_\ell^{k+1} \leftarrow X_\ell^k \cap \{x : \pi^\top x \geq \pi_0\}$.
5:   $k \leftarrow k + 1$.
6: **end while**

A natural question that arises regarding a cutting-plane algorithm is whether it converges to an optimal solution of the MIP (if it exists) in finitely many steps. This question is important not only in the context of mixed-integer programming, but also for designing algorithms for stochastic mixed-integer programs. In this article, we survey cutting-plane algorithms for subclasses of mixed-integer programs and discuss the key results needed to show their convergence for different classes of MIPs. We also give examples to show non-convergence when applied to other classes of problems. We begin with a discussion of the Gomory cutting-plane method and its convergence for pure integer programs.

2 Algorithms using Lexicography and Rounding

First, we discuss the Gomory cutting-plane algorithm [23], which is one of the first cutting-plane algorithms developed to solve pure integer programming problems ($p = n$ in (1)). Gomory cuts and extensions are discussed elsewhere in this encyclopedia. (Please refer to the encyclopedia article by R. Fukasawa.)

We make the following assumptions:

- The data of the mixed-integer program are integral, i.e., $c \in \mathbb{Z}^n, A \in \mathbb{Z}^{m \times n}, b \in \mathbb{Z}^m$. This assumption is without loss of generality because the original rational coefficients can be appropriately scaled to obtain an equivalent representation of X using integer coefficients.
- The problem is stated equivalently by minimizing the variable x_0 and adding the constraint $x_0 = c^\top x$ to the constraint set.

For the sake of completeness, we derive the Gomory fractional cut in this article. Let $\bar{x}$ be the optimal solution to the LP relaxation (2), let $\mathcal{B}$ and $\mathcal{N}$ denote the index set of basic and nonbasic variables associated with this optimal solution, respectively, and let B and N denote the columns of basic and nonbasic variables, respectively. If $\bar{x} \in \mathbb{Z}^n_+$, then we have found an optimal solution to the MIP. Otherwise, we choose a row of the simplex tableau corresponding to a variable $i \in \mathcal{B}$ such that $\bar{x}_{\mathcal{B}(i)} \notin \mathbb{Z}$:

$$x_{\mathcal{B}(i)} + \sum_{j \in \mathcal{N}} \bar{a}_{ij} x_j = \bar{b}_i, \tag{3}$$

where $\bar{a}_{ij}$ is a component of the matrix $B^{-1}N$, $\mathcal{B}(i)$ is the ith basic variable and $\bar{b}_i := \bar{x}_{\mathcal{B}(i)}$. Because $x_j \geq 0$ for all j, we can write

$$x_{\mathcal{B}(i)} + \sum_{j \in \mathcal{N}} \lceil \bar{a}_{ij} \rceil x_j \geq \bar{b}_i.$$

Furthermore, because $x_j \in \mathbb{Z}$ for all j, we have,

$$x_{\mathcal{B}(i)} + \sum_{j \in \mathcal{N}} \lceil \bar{a}_{ij} \rceil x_j \geq \lceil \bar{b}_i \rceil. \tag{4}$$

Subtracting (3) from (4), we get the Gomory fractional cut as

$$\sum_{j \in \mathcal{N}} \phi(\bar{a}_{ij}) x_j \geq \phi(\bar{b}_i), \tag{5}$$

where $\phi(\beta) := \lceil \beta \rceil - \beta$. Note that the current basis and solution is cut off by (5). An iteration of Gomory's cutting-plane algorithm involves solving a linear program associated with the current iteration. If the solution is fractional, then a cut of the form (5) is generated, which is guaranteed to cut off this fractional solution.

The proof of convergence of Gomory's fractional cutting-plane algorithm relies on the lexicographic dual simplex method [25, 30, 33] (see the article by M. Banciu in the encyclopedia for a description of the lexicographic dual simplex method). Given two vectors $v^1, v^2 \in \mathbb{R}^n$, v^1 is said to be *lexicographically larger* than v^2, denoted as $v^1 \succ v^2$, if there exists a k such that $v^1_k > v^2_k$ and $v^1_j = v^2_j$ for all $j = 1, \ldots, k-1$. We say that v^1 is lexicographically larger than or equal to v^2, denoted by $v^1 \succeq v^2$, if either $v^1 \succ v^2$ or $v^1 = v^2$. In the lexicographic dual simplex method, one begins with and maintains simplex tableaus that are lexicographically dual feasible, i.e., tableaus that are dual feasible and have all the nonbasic columns lexicographically smaller than the zero vector. This is accomplished by using a lexicographic pivoting rule. This approach ensures that the simplex method does not cycle, and furthermore, guarantees that the lexicographically smallest optimal solution to the linear program, if it exists, is found in *finitely many steps*.

In line 3 of Algorithm 1, when one uses the Gomory fractional cutting-plane method for pure integer programs, one solves the kth relaxation using the lexicographical dual simplex method. Because $X^{k+1}_\ell \subset X^k_\ell$ for each iteration k with a fractional solution x^k, we obtain a sequence of lexicographically increasing solutions $\{x^t\}^k_{t=0}$ during the execution of the cutting-plane procedure. What remains to be seen is whether this sequence converges to an integer solution in finitely many steps.

It is a well-known fact that if $\hat{x}$ is an extreme point of $\text{conv}(X)$, then there exists $M < \infty$ such that $\hat{x}_j \leq M$ with $M \in \mathbb{Z}$ for $j = 1, \ldots, n$ (see Nemhauser and Wolsey [30], Theorem 4.1 in Chapter I.5). That is, the extreme point components of the convex hull of an integer set within a polyhedron are bounded. Moreover the bound depends only on the problem data. Let $\beta^- := \min(\beta, 0)$ and $\beta^+ := \max(0, \beta)$. From the preceding fact, it follows that if $x \in X$, then

$$\left(\sum_{j=1}^n c_j^+ M, M, \ldots, M \right)^\top \succeq x \succeq \left(\sum_{j=1}^n c_j^- M, 0, \ldots, 0 \right)^\top. \tag{6}$$

The following result establishes the fact that Gomory cuts, when used with the lexicographic dual simplex method and when cuts are chosen from the smallest index variable that is fractional in the solution, have a certain "rounding" property.

Proposition 1 ([33])**.** *Let x^t denote the lexicographically smallest optimal solution to $\min_{x \in X_\ell^t} c^\top x, t = k-1, k$, where the updated approximation X_ℓ^k is formed by adding to X_ℓ^{k-1} a Gomory cut (5) corresponding to the smallest index source row of the fractional solution x^{k-1}. Let i_k denote the index of the variable used to form the Gomory cut, and define*

$$\alpha^k := \left(x_0^{k-1}, x_1^{k-1}, \ldots, x_{i_k-1}^{k-1}, \lceil x_{i_k}^{k-1} \rceil, 0, \ldots, 0\right)^\top. \tag{7}$$

Then,

$$x^k \succeq \alpha^k \succ x^{k-1}.$$

It is important to note that the fractional variable with the smallest index is chosen for cut generation. Finite convergence of the Gomory cutting-plane method follows because from Proposition 1 we obtain a sequence of lexicographically increasing vectors at each iteration, and there are only finitely many integer vectors that satisfy (6).

Theorem 1 ([30, 33])**.** *Suppose that at every iteration of the Gomory cutting-plane method, the cut is formed by choosing smallest index source row of the fractional solution, and the lexicographic dual simplex method is used to optimize the updated problems. Then, the cutting-plane method finds an optimal solution or concludes that the problem is infeasible in finitely many steps.*

Note that as long as the Gomory cut from the smallest index source row is present, lexicography allows other valid inequalities to be added to the subsequent approximation X_ℓ^{k+1} of the Gomory cutting-plane method without losing finite convergence. For example, one can add as many Gomory cuts as there are fractional variables in the solution to the current approximation. This approach of adding cuts corresponding to all fractional variables in the linear relaxation is referred to as adding a *round of cuts*.

For the Gomory cutting-plane algorithm, the key elements needed to show convergence are lexicography and the choice of the source row for cut generation. The nature of convergence can be interpreted as pruning nodes in a lexicographic enumeration tree (c.f. [30]). In this tree, the nodes at level j correspond to an index of a variable. For example, the root node corresponds to the objective function variable x_0 and there are as many branches as there are possible values of the variable. All possible integer vectors satisfying (6) are arranged in a lexicographic order. In an iteration k, any vector in the tree that is strictly lexicographically smaller than α^k can be thought of as pruned. Thus, by means of this tree, lexicography implicitly provides a "memory" of vectors pruned. We shall see that the concept of "memory" is a recurrent theme in the convergence proofs of various cutting-plane algorithms throughout this article.

Lexicography has also been used in other finitely convergent cutting-plane algorithms. Neto [32] proposes an iterative lexicographic optimization method for the solution of mixed-integer programs. The author shows that under the assumptions that the optimal objective function is integral and bounded, and the integer variables are bounded, the algorithm converges in finitely many iterations to the lexicographically smallest optimal solution. Unlike Gomory's cutting-plane method, the cutting planes in this algorithm are not derived from the simplex tableau of the LP relaxation. They are not necessarily valid for the convex hull of solutions to the mixed-integer program, but they cut off a fractional linear programming solution, while ensuring that any lexicographically smaller integer feasible solution is not cut off. This algorithm is a generalization of the algorithm given in [35], which gives a finite cutting-plane algorithm for mixed-binary problems with integral optimal objective function values. Other finitely convergent algorithms for pure integer programs are proposed in [2, 9, 11].

Gomory also proposed the Gomory mixed integer (GMI) cuts [24] for general MIP problems. When the objective function value is known to be integer, the proof for the pure integer case can be adapted to show finite convergence for the mixed-integer case. However, if the objective function value is no longer required to be integer, a cutting-plane algorithm based on GMI cuts may not converge. The following example [43] (see also Padberg [37]) is an instance on which GMI cuts do not converge.

$$
\begin{aligned}
&\min -x_3 \\
&-x_1 - x_2 - x_3 \geq -2 \\
&x_1 - x_3 \geq 0 \\
&x_2 - x_3 \geq 0 \\
&x_1, x_2 \in \mathbb{Z}_+, x_3 \in \mathbb{R}_+.
\end{aligned}
\tag{8}
$$

One can easily observe that $x_3 = 0$ in an optimal solution to this example. When one does not consider the objective function row corresponding to x_3 for cut generation, Padberg [37] shows that the optimal solution at the kth iteration of the GMI cutting-plane method (when a round of cuts is added in every iteration) is $x_1^k = x_2^k = (2k+2)/(2k+3), x_3^k = 2/(2k+3)$. Thus, for any iteration k one has $x_3 > 0$ and one only obtains an optimal solution as $k \to \infty$.

Although Gomory fractional and mixed integer cuts were developed in the late 1950's and early 1960's, they were not widely used until the late 1990's. Balas et al. [7] demonstrated the computational effectiveness of Gomory cuts when used within a branch-and-cut scheme. More recently, Zanette et al. [44] showed that the performance of a pure Gomory fractional cutting-plane method is far superior when the lexicographic dual simplex method is used instead of the regular dual simplex. In their implementation, instead of using the lexicographic pivoting rule in the dual simplex, they use a post-processing scheme to obtain the lexicographically smallest solution. Other computational enhancements are reported in [8].

3 Algorithms using Sequential Convexification

3.1 Lift-and-Project Cutting-Plane Algorithm

In this section, we discuss the lift-and-project cutting-plane algorithm [5] for mixed binary programs, i.e., the mixed-integer restrictions in (1) are $x \in \{0,1\}^p \times \mathbb{R}_+^{n-p}$. The lift-and-project algorithm has its roots in disjunctive programming. Disjunctive programming was introduced by Balas [3] and refers to optimization over union of polyhedra. Disjunctions arise naturally in integer programming. For example a binary restriction on a variable $x_j \in \{0,1\}$ can be represented as a disjunction $(x_j = 0) \vee (x_j = 1)$, where $\vee$ denotes the logical "or" operator. Similarly, an integer variable $x_j \in [0, U_j] := \{0, 1, \ldots, U_j\}$ can be represented using the disjunction $(x_j = 0) \vee (x_j = 1) \vee \cdots \vee (x_j = U_j)$. A variety of disjunctions can be used to generate cuts for an MIP. For example, when the LP relaxation of an MIP yields a solution $\bar{x}$ with $\bar{x}_j$ fractional for $j \in \{1, \ldots, p\}$, a disjunction of the type $(X_\ell \cap \{x_j \leq \lfloor \bar{x}_j \rfloor\}) \vee (X_\ell \cap \{x_j \geq \lceil \bar{x}_j \rceil\})$ can be written to produce a cut that separates this point from $\mathrm{clconv}(X)$. When the disjunction used for cut generation contains only two terms of the type mentioned above, we refer to such cuts as elementary disjunctive cuts. More general two-term disjunctions (e.g. *split disjunctions* [17]), disjunctions with more than two terms (multi-term disjunctions) and so forth can be considered to generate violated inequalities. We refer the reader to the articles in this encyclopedia by L. Liberti, Q. Louveaux and K. Andersen and the references therein for further details.

The lift-and-project algorithm relies on a sequential convexification property of mixed binary programs. Mixed binary programs inherit this property from a more general class of problems called facial disjunctive programs (c.f. [4]). Given a general mixed-integer program (1), the convexification with respect to a variable $x_j, j \in \{1, \ldots, p\}$ and a polyhedron Y is defined as:

$$P_j(Y) := \text{clconv}(Y \cap \{x : x_j \in \mathbb{Z}\}).$$

Let $i_1 \ldots, i_t$, where $i_j \in \{1, \ldots, p\}, j = 1, \ldots, t$ be variable indices. For $t \geq 2$ define

$$P_{i_t,\ldots,i_1}(X_\ell) := P_{i_t}(P_{i_{t-1}}(\cdots(P_{i_1}(X_\ell))\cdots)),$$

i.e., first, the mixed-integer set defined by imposing integrality of x_{i_1} to X_ℓ is convexified. This results in the set $P_{i_1}(X_\ell)$. Then, the set defined by imposing integrality of x_{i_2} to $P_{i_1}(X_\ell)$ is convexified, and so forth. Imposing integrality variable-by-variable and taking the closure of the convex hull at each step is referred to as *sequential convexification* in the sequence defined by the permutation $i_1, \ldots, i_p$.

For mixed binary programs, when the sets are convexified with respect to the binary variables sequentially in the manner described above, Balas [3] showed that after p steps, one obtains the closure of the convex hull of X, i.e., $P_{i_p,\ldots,i_1}(X_\ell) = \text{clconv}(X)$. Balas [3] also gave an example showing that for general mixed-integer programs (with $x \in \mathbb{Z}^p_+ \times \mathbb{R}^{n-p}_+$) sequential convexification may fail to yield $\text{clconv}(X)$. Note that general mixed-integer programs are not a subclass of facial disjunctive programs.

Theorem 2 ([3, 5])**.** *For mixed binary programs*

$$P_{i_p,\ldots,i_1}(X_\ell) = \text{clconv}(X).$$

Sherali and Adams [40, 41] give an alternative finite characterization of the convex hull of the binary and mixed-binary integer programs based on the reformulation-linearlization technique (RLT). We refer the reader to the encyclopedia article by H. D. Sherali for further details. See also [28] for another finite construction of the convex hull.

Assume that the constraints $Ax \geq b$ in the mixed-binary program include the constraints $x \leq 1$. In the lift-and-project cutting-plane method, cut generation is a two-step process. The first step is a lifting step, which involves representing the convex hull of the disjunctive program as a linear program in a higher dimensional space (of dimension larger than n). We shall consider the constructions in [5]. If the solution to the linear relaxation is fractional, with x_j fractional, then one first constructs the nonlinear system

$$\begin{aligned}&(1 - x_j)(Ax - b) \geq 0\\&x_j(Ax - b) \geq 0.\end{aligned}$$

This system is linearized by setting $x_i x_j =: z_i, i \neq j$ and $x_j^2 = x_j$ (the latter identity holds because x_j is binary). The new linear system has additional variables; let its feasible set be denoted by $M_j(X_\ell)$. Alternatively, one can write a disjunction

$$D_j := (X_\ell \cap \{x_j = 0\}) \vee (X_\ell \cap \{x_j = 1\}). \tag{9}$$

Consider the linear program,

$$
\begin{aligned}
&\min c^\top x \\
&Ay^1 - by_0^1 \geq 0 \\
&Ay^2 - by_0^2 \geq 0 \\
&-y_j^1 = 0 \\
&y_j^2 - y_0^2 = 0 \\
&y_0^1 + y_0^2 = 1 \\
&x = y^1 = y^2. \\
&x, y^1, y^2 \geq 0, y_0^1, y_0^2 \geq 0,
\end{aligned}
$$

and let $K_j(X_\ell)$ denote the feasible region of this linear program.

The second step of the lift-and-project algorithm is to project the higher dimensional linear program onto the space of the original variables. Balas et al. [5] show that the projection of $M_j(X_\ell)$ (equivalently, the projection of $K_j(X_\ell)$) onto the space of x variables yields $P_j(X_\ell)$. Balas et al. [5] show that valid inequalities for $P_j(X_\ell)$ correspond to feasible points of projection cone of $M_j(X_\ell)$. If X_ℓ is full dimensional, the facets of $P_j(X_\ell)$ correspond to the extreme rays of the projection cone of $M_j(X_\ell)$. These results are specialized versions of the more general result on the compact representation of the closure of the convex hull of a disjunctive program in a higher dimensional space [4]. In order to produce violated lift-and-project cuts for a fractional solution to the linear relaxation, one writes the so-called cut generating linear program (CGLP), that is formulated by writing the projection cone of $M_j(X_\ell)$ (or $K_j(X_\ell)$) and adding a normalization constraint that truncates the cone (see also [42]). In addition, one adds an objective function, for example, an objective function that maximizes the violation of the cut.

In a cutting-plane algorithm that employs lift-and-project cuts for mixed binary programs, one can always generate a violated inequality for a fractional solution by forming a disjunction. However, as shown in an example in Sen and Sherali [39], a naive implementation of the lift-and-project cutting-plane algorithm may not be finitely convergent. They show that appropriate memory needs to be maintained to guarantee finite convergence. The main idea behind the convergence proof of the lift-and-project algorithm is to carefully select the fractional variables and polyhedra for generating cuts that lead to sequential convexification of the linear relaxation with respect to the binary variables. To this end, let a j-cut be a valid inequality for $P_j(\cdot)$ obtained by solving a CGLP with an appropriate normalization. Also let S_i^k be the polyhedron defined by the inequalities $Ax \geq b$ and all j-cuts for $j = 1, \ldots, i$ at iteration k, and let S_0^k denote the linear relaxation X_ℓ. In line 3 of Algorithm 1, to generate a cut, one chooses the *largest index* $j \in \{1, \ldots, p\}$ such that x_j is fractional, and generates an inequality for $P_j(S_{j-1}^k)$ using the CGLP with an appropriate normalization constraint. One can show that the cut generated for $P_j(S_{j-1}^k)$ cuts off the fractional solution x^k. To show finite convergence, observe that because the projection cone of $M_1(S_0^k) = M_1(X_\ell)$ has finitely many extreme points, this set is convexified with respect to x_1 after the addition of finitely many 1-cuts. By induction, one can show that the number of j-cuts generated is finite for all $j = 1, \ldots, p$. Thus, in the worst case, we will have constructed $\text{clconv}(X)$ in finitely many steps, at which point an optimal solution (if it exists) is found. Let m_j be the iteration index when the last i-cut was generated for $i = 1, \ldots, j$. Note that $S_j^k = S_j^{m_j}$ in all iterations $k \geq m_j$. So the iteration index m_j provides the memory of inequalities defining S_j^k for each j, and it plays an important role in the convergence proof. We refer the reader to Sen and Sherali [39] for a related proof of convergence, and [10, 26] for an alternative finitely convergent implementation of a pure cutting-plane algorithm for facial disjunctive programs.

Theorem 3 ([5]). *Suppose at each iteration k the lift-and-project cutting-plane algorithm is implemented* *by selecting the largest index j that is fractional and by generating a cut corresponding to an extreme point of the CGLP (with appropriate normalization) formed using the polyhedron S^k_{j-1}. Then, the cutting-plane method finds an optimal solution or concludes that the problem is infeasible in finitely many steps.*

The first computational results incorporating lift-and-project cuts within a branch-and-cut algorithm are reported in [6]. We refer the reader to the encyclopedia article by Q. Louveaux for a survey of recent computational developments.

3.2 Cutting-Plane-Tree Algorithm

As discussed earlier, a variable-by-variable sequential convexification may not yield the convex hull of feasible solutions for general mixed-integer programs. In fact, Owen and Mehrotra [36] give the following example, showing that when the lift-and-project cutting-plane algorithm is applied to the mixed-integer case one may not converge to an optimal solution even in the limit.

$$\begin{aligned}
&\min -x_1 - x_2\\
&\text{s.t. } 8x_1 + 12x_2 \le 27\\
&8x_1 + 3x_2 \le 18\\
&0 \le x_1, x_2 \le 3\\
&x \in \mathbb{Z}^2_+.
\end{aligned}$$

It can be easily verified that the convex hull of the feasible set of the MIP given above is $\{x \in \mathbb{R}^2_+ : x_1 + x_2 \le 2\}$. However, an application of the lift-and-project algorithm by using elementary (two-term) disjunctions of the type $(X_\ell \cap \{x_j \le \lfloor \bar{x}_j \rfloor\}) \vee (X_\ell \cap \{x_j \ge \lceil x_j \rceil\})$ yields in the limit the set $\{x \in \mathbb{R}^2_+ : x_1 + x_2 \le 2.25\}$.

In a recent paper, Chen et al. [14] propose the cutting-plane-tree (CPT) algorithm, which uses cuts derived from multi-term disjunctions. The multi-term disjunctions partition the feasible region into a finite collection of intervals for each variable, the union of which gives contains the feasible domain. Iteratively finer partitions are obtained over the course of the algorithm.

We first review a convexification result for general MIP due to Chen et al. [14]. Assume that each integer variable is bounded and can take integer values in the interval $[0, U_j]$. Let each interval be further divided into sub intervals

$$[L_{1j} := 0, U_{1j}], [L_{2j}, U_{2j}], \dots [L_{t_j j}, U_{t_j j} := U_j],$$

where $L_{\kappa_j j}, U_{\kappa_j j} \in \mathbb{Z}_+, L_{\kappa_j j} \le U_{\kappa_j j}$ and $\kappa_j \in \{1, \dots, t_j\}, j = 1, \dots, p$ with $L_{\kappa_j+1 j} - U_{\kappa_j j} \le 1$ for $\kappa_j \in 1, \dots, t_j - 1$ so that the subintervals span all integers in $[0, U_j]$. Given a such partition $\mathcal{P}$, the collection of all p tuples of the form $\kappa = (\kappa_1, \kappa_2, \dots, \kappa_p)$ is denoted by $K(\mathcal{P})$. A *unit partition* $\mathcal{P}^*$ is a partition where $U_{\kappa_j j} - L_{\kappa_j j} \le 1$ for all $\kappa_j \in \{1, \dots, t_j\}, j = 1, \dots, p$. For $\kappa \in K(\mathcal{P}^*), j \in \{1, \dots, p\}$ and a polyhedron $\bar{X}$, we define the following sets:

$$\mathcal{P}^-(\kappa, j, \bar{X}) := \{x \in \bar{X} : L_{\kappa_i i} \le x_i \le U_{\kappa_i i}, i = 1, \dots, p; x_j \le L_{\kappa_j j}\}$$

and,

$$\mathcal{P}^+(\kappa, j, \bar{X}) := \{x \in \bar{X} : L_{\kappa_i i} \le x_i \le U_{\kappa_i i}, i = 1, \dots, p; x_j \ge U_{\kappa_j j}\}.$$

In addition, define $\mathcal{H}^\kappa_j(\bar{X}) := \text{clconv}(\mathcal{P}^-(\kappa, j, \bar{X}) \cup \mathcal{P}^+(\kappa, j, \bar{X}))$. The following theorem gives a sequential convexification result for a general MIP with respect to a unit partition $\mathcal{P}^*$.

Theorem 4 ([14])**.** *Assume that the set X as defined in (1) is nonempty and has bounded integer* *variables. Then for any unit partition $\mathcal{P}^*$,*

$$\text{clconv}(X) = \text{clconv}\left\{\cup_{\kappa\in K(\mathcal{P}^*)}[\mathcal{H}_p^\kappa(\mathcal{H}_{p-1}^\kappa(\cdots(\mathcal{H}_1^\kappa(X_\ell))\cdots))\setminus\emptyset]\right\}.$$

Adams and Sherali [1] propose an alternative finite convex hull characterization for general MIP based on Lagrange interpolation polynomials. Similar to the case with mixed binary programs described in Section 3.1, the existence of a finite convex hull representation suggests the existence of a finite pure cutting-plane algorithm. However, care must be taken in the design of the algorithm to guarantee convergence. Next we describe such an algorithm, referred to as the cutting-plane-tree (CPT) algorithm [14].

In the cutting-plane tree $\mathcal{T}$, there is a single root node o. For each node $\sigma \in \mathcal{T}$, an integer $v_\sigma \in \{1, 2, \ldots, n_1\}$ stores the index of the integer variable that is split, an integer q_σ stores the (lower) level of the splitting. Let l_σ, r_σ and p_σ denote the left child, right child and parent nodes of node σ, respectively. Let $\mathcal{S}(\sigma)$ be all nodes on the subtree rooted at node σ (not including node σ and the leaf nodes). Let $\mathcal{N}(\sigma)$ be the collection of the nodes on the path from the root node to node σ (not including the root node), let $\mathcal{N}^-(\sigma)$ be the collection of nodes in $\mathcal{N}(\sigma)$ that were formed as the left child node of its parent, and let $\mathcal{N}^+(\sigma)$ be the collection of nodes in $\mathcal{N}(\sigma)$ that were formed as the right child node of its parent. Given $\sigma \in \mathcal{T}$ define

$$\mathcal{C}_\sigma = \{x | x_j \in [0, U_j], x_{v_{p_s}} \le q_{p_s}, \forall s \in \mathcal{N}^-(\sigma), x_{v_{p_s}} \ge q_{p_s} + 1, \forall s \in \mathcal{N}^+(\sigma)\}.$$

We let m_σ store an iteration index, which keeps track of the cutting planes that will be used to generate a disjunctive cut when this node is revisited. In other words, m_σ determines the set X^{m_σ} to be used in the cut generation LP (CGLP) [3, 42]. The set X^{m_σ} corresponds to X_ℓ together with the first $m_\sigma - 1$ cuts added to it. If $X^{m_\sigma} \cap \mathcal{C}_{l_\sigma} = \emptyset$ ($X^{m_\sigma} \cap \mathcal{C}_{r_\sigma} = \emptyset$), we say that the left (right) child node of σ is "fathomed", i.e., $l_\sigma = null$ ($r_\sigma = null$).

At iteration k of the CPT algorithm, if the current extreme point solution to $\min_{x\in X^k} c^T x$, given by x^k is integral, then we have found an optimal solution to the MIP. Otherwise, we search the cutting-plane tree, to find the last node σ on the path from the root node such that $x^k \in \mathcal{C}_\sigma$. There are two cases: Case (1) σ is a leaf node ($\sigma \in \mathcal{L}_k$, where $\mathcal{L}_k$ is the set of leaf nodes of the CPT at iteration k), Case (2) σ is not a leaf node ($\sigma \notin \mathcal{L}_k$, $x^k \in \mathcal{C}_\sigma$, $x^k \notin \mathcal{C}_{l_\sigma}$ and $x^k \notin \mathcal{C}_{r_\sigma}$). In Case (1), we choose a fractional variable x_j, $j = 1, \ldots, n_1$ with the smallest index, and let the split variable be $v_\sigma = j$. We create two new nodes: left (l_σ) and right (r_σ) children of σ at the split level $q_\sigma = \lfloor x_j^k \rfloor$. We let $\mathcal{C}_{l_\sigma} = \{x \in \mathcal{C}_\sigma | x_j \le \lfloor x_j^k \rfloor\}$ and $\mathcal{C}_{r_\sigma} = \{x \in \mathcal{C}_\sigma | x_j \ge \lceil x_j^k \rceil\}$. In this case, we also let $m_\sigma = k$, as this is the first time the tree search for a fractional solution stops at node σ. Let $\mathcal{L}_{k+1} \leftarrow (\mathcal{L}_k \cup \{l_\sigma, r_\sigma\}) \setminus \{\sigma\}$. In Case (2), the cutting-plane tree and m_σ are unchanged, i.e., $\mathcal{L}_{k+1} \leftarrow \mathcal{L}_k$. However, in this case, we update $m_t = k$ for all successors of σ, $t \in \mathcal{S}(\sigma)$. We generate a valid inequality for the set $\text{clconv}\{\cup_{t\in\mathcal{L}_{k+1}}(X^{m_\sigma} \cap \mathcal{C}_t)\}$ that cuts off x^k (from an extreme point of the associated CGLP). The new inequality is included along with those defining X^k, and the resulting set is denoted X^{k+1}. This process continues until an optimal solution to the MIP is obtained.

Theorem 5 ([14])**.** *Assume that the set X is non-empty and has bounded integer variables. Then, the cutting-plane-tree algorithm converges to an optimal solution of the MIP in finitely many iterations.*

The convergence proof relies on the tree data structure and the iteration index m_σ that serves as memory for storing the subset of cuts generated until that iteration of the algorithm. A careful reader will recognize the similarity between the memory index m_j in Section 3.1 with the memory index m_σ introduced in this section.

Table 1 shows the first three iterations of the execution of the cutting-plane-tree algorithm on the example of Owen and Mehrotra [36] that was presented at the beginning of this section. In this table, k,

x^k, σ, m_σ denote the iteration number, solution of the relaxation, the node and the memory parameter, respectively. The sets Q_i in an iteration represent a subset of $[0,3] \times [0,3]$ in which the current solution can fall. We begin the algorithm by solving the linear relaxation corresponding to the root node $\sigma = 1$ (setting $m_1 = 1$) and we obtain a fractional solution $(15/8, 1)$. We create two child nodes for 1, namely 2 and 3 representing $x_1 \leq 1$ and $x_1 \geq 2$, respectively. We then generate a disjunctive cut using the CGLP corresponding to the disjunction defined by Q_1 and Q_2 on X_ℓ, add it to the linear relaxation and re-solve the problem to get $x^2 = (2, 2/3)$. Because x^2 satisfies the bounds specified by node 3, and node 3 is a leaf node, we set $m_3 = 2$ and we create two child nodes for node 3. Because the set $X_\ell \cap \{x : 2 \leq x_1 \leq 3, 1 \leq x_2 \leq 3\}$ corresponding to the right child of 3 is infeasible, we remove it from further consideration. The CGLP in iteration 2 is constructed by using X_ℓ, cut 1 (because $m_3 = 2$) and the disjunctions defined by $Q_i, i = 1, 2$ and a disjunctive cut is generated. In the third iteration, upon solving the updated relaxation, we get another fractional solution $(1, 19/12)$, which falls at node 2, which is a leaf node. So we set $m_2 = 3$ and create two child nodes for node 2. It turns out that the right node of node 2 is infeasible, so we discard it from further consideration and we are left with the sets $Q_i, i = 1, \dots 3$. The algorithm proceeds in this manner and generates the cut $x_1 + x_2 \leq 2$ in the sixth iteration (see [14]). In the subsequent solution of the linear relaxation, we find an optimal integer solution and the algorithm terminates. Note that while the convex hull was generated at termination in this example, this generally need not be the case for the algorithm to terminate.

Table 1: First three iterations of the CPT algorithm on Owen and Mehrotra's [36] example

k	x^k	σ	m_σ	$\mathcal{Q}_{k+1}$	cut
1	$(15/8, 1)$	1	1	$Q_1 = [0,1] \times [0,3]$ $Q_2 = [2,3] \times [0,3]$	$11/12x_1 + x_2 \leq 5/2$
2	$(2, 2/3)$	3	2	$Q_1 = [0,1] \times [0,3]$ $Q_2 = [2,3] \times [0,0]$	$x_1 + 15/19x_2 \leq 9/4$
3	$(1, 19/12)$	2	3	$Q_1 = [2,3] \times [0,0]$ $Q_2 = [0,1] \times [0,1]$ $Q_3 = [0,1] \times [2,3]$	$x_1 + 15/16x_2 \leq 9/4$

Computational results with a branch-and cut algorithm based on cuts from the cutting-plane-tree are presented in [15]. Jörg [27] proposes another finitely convergent pure cutting-plane algorithm for general MIP using multi-term disjunctions in conjunction with Gomory cuts. An iteration of this approach involves the identification of all alternative optimal solutions of a related linear program and the enumeration of all extreme rays of a polyhedral projection cone.

4 Conclusions

In this paper, we survey different pure cutting-plane algorithms and the key results needed to show their convergence. Throughout, we emphasize the important role that memory plays in the convergence proofs. Various studies indicate that pure cutting plane algorithms are not practical for deterministic MIPs, but cutting planes are crucial for the solution of difficult MIPs in a branch and cut algorithm (see the paper by J. Mitchell in the encyclopedia for an introduction to branch and cut algorithms). However, pure cutting-plane algorithms have important applications in the solution of two-stage stochastic MIPs. We refer the reader to the papers by Sen and Higle [38] for a decomposition algorithm based on lift-and-project cuts for two-stage stochastic mixed-binary programs, to Gade et al. [22] for decomposition algorithms using Gomory cuts for two-stage stochastic pure integer programs, and to Ntaimo [34] for a decomposition algorithm for two-stage stochastic mixed-binary programs based on the so-called Fenchel cuts. (A pure cutting-plane algorithm based on Fenchel cuts [12] is shown to be finitely convergent for deterministic MIPs in [13]. The separation of Fenchel cuts involves the solution of a MIP.)

An area closely related to the convergence of pure cutting-plane algorithms is the study of the convergence of *closures* of valid inequalities in mixed-integer programs. Given a polyhedron and a family of valid inequalities, the closure of the family of valid inequalities with respect of the polyhedron is obtained by adding all valid inequalities in the family to the polyhedron. Note that the number of inequalities in a given family can be infinite. For example, the family of Chvátal inequalities (which are equivalent to Gomory fractional cuts, not necessarily constructed from a basis of the LP relaxation) are of the type $\lceil u^\top A \rceil \geq \lceil u^\top b \rceil$ for $u \in \mathbb{R}^m_+$ and the vector ceiling is performed component-wise. Chvátal [16] shows that for pure integer programs, the Chvátal closure is a polyhedron, i.e., only finitely many Chvátal inequalities are needed to describe the closure. Moreover, starting with the linear relaxation of a pure integer program, taking Chvátal closures recursively yields the convex hull in finitely many steps. For mixed-integer programs, Cook et al. [17] show that the split closure is a polyhedron. However, using an example similar to the one given in this article in Section 2, they show that taking the split closure recursively does not yield the convex hull in finitely many steps. Nemhauser and Wolsey [31] show the equivalence between Gomory mixed integer cuts, mixed integer rounding cuts and split cuts and as a result, a cutting-plane algorithm based on any of these classes is not finitely convergent for general mixed-integer programs. Recently, Del Pia and Weismantel [21] show that for mixed-integer programs, the split closure converges to the convex hull in the limit. They also show that the closure of a certain family of cuts called lattice-free cuts is a polyhedron and that taking this closure recursively yields the convex hull in finitely many steps for general mixed-integer programs. Connections between lattice free cuts and general multi-term disjunctive cuts are studied in [20], who also give a procedure to construct the closure of the convex hull in finitely many steps. The relationships amongst the closures of various families of valid inequalities are studied in [19].

Acknowledgments

The authors thank Suvrajeet Sen for many discussions on convergence of cutting planes and their applications in decomposition algorithms for stochastic integer programs. This work was supported, in part, by NSF-CMMI Grant 1100383.

References

1. W. P. Adams and H. D. Sherali, *A hierarchy of relaxations leading to the convex hull representation for general discrete optimization problems*, Annals of Operations Research, 140 (2005), pp. 21–47.

2. R. Armstrong, A. Charnes, and F. Phillips, *Page cuts for integer interval linear programming*, Discrete Applied Mathematics, 1 (1979), pp. 1 – 14.

3. E. Balas, *Disjunctive programming*, in Discrete Optimization II, P. Hammer, E. Johnson, and B. Korte, eds., vol. 5 of Annals of Discrete Mathematics, Elsevier, 1979, pp. 3–51.

4. ———, *Disjunctive programming: Properties of the convex hull of feasible points*, Discrete Applied Mathematics, 89 (1998), pp. 3–44.

5. E. Balas, S. Ceria, and G. Cornuéjols, *A lift-and-project cutting plane algorithm for mixed 0–1 programs*, Mathematical Programming, 58 (1993), pp. 295–324.

6. ———, *Mixed 0-1 programming by lift-and-project in a branch-and-cut framework*, Management Science, (1996), pp. 1229–1246.

7. E. Balas, S. Ceria, G. Cornuéjols, and N. Natraj, *Gomory cuts revisited*, Operations Research Letters, 19 (1996), pp. 1–9.

8. E. Balas, M. Fischetti, and A. Zanette, *On the enumerative nature of Gomory's dual cutting plane method*, Mathematical Programming, 125 (2010), pp. 325–351.

9. D. Bell, *A cutting plane algorithm for integer programs with an easy proof of convergence*, Tech. Rep. WP–73–15, International Institute for Applied Systems Analysis, Laxenburg, 1973.

10. C. E. Blair, *Facial disjunctive programs and sequence of cutting-planes*, Discrete Applied Mathematics, 2 (1980), pp. 173 – 179.

11. V. J. Bowman and G. L. Nemhauser, *A finiteness proof for modified Dantzig cuts in integer programming*, Naval Research Logistics Quarterly, 17 (1970), pp. 309–313.

12. E. A. Boyd, *Fenchel cutting planes for integer programs*, Operations Research, 42 (1994), pp. 53–64.

13. ———, *On the convergence of Fenchel cutting planes in mixed-integer programming*, SIAM Journal on Optimization, 5 (1995), pp. 421–435.

14. B. Chen, S. Küçükyavuz, and S. Sen, *Finite disjunctive programming characterizations for general mixed-integer linear programs*, Operations Research, 59 (2011), pp. 202–210.

15. ———, *A computational study of the cutting plane tree algorithm for general mixed-integer linear programs*, Operations Research Letters, 40 (2012), pp. 15–19.

16. V. Chvátal, *Edmonds polytopes and a hierarchy of combinatorial problems*, Discrete Mathematics, 4 (1973), pp. 305–337.

17. W. Cook, R. Kannan, and A. Schrijver, *Chvátal closures for mixed integer programming problems*, Mathematical Programming, 47 (1990), pp. 155–174.

18. G. Cornuéjols, *Valid inequalities for mixed integer linear programs*, Mathematical Programming, 112 (2008), pp. 3–44.

19. G. Cornuéjols and Y. Li, *Elementary closures for integer programs*, Operations Research Letters, 28 (2001), pp. 1–8.

20. S. Dash, N. Dobbs, O. Günlük, T. Nowicki, and G. Swirszcz, *Lattice-free sets, branching disjunctions, and mixed-integer programming.* IBM Research Report http://researcher.watson.ibm.com/researcher/files/us-sanjeebd/purecuts-oo.pdf, 2012.

21. A. Del Pia and R. Weismantel, *On convergence in mixed integer programming*, Mathematical Programming, 135 (2012), pp. 397–412.

22. D. Gade, S. Küçükyavuz, and S. Sen, *Decomposition algorithms with parametric Gomory cuts for two-stage stochastic integer programs*, Mathematical Programming, (2012). Article in Advance, DOI: 10.1007/s10107-012-0615-y.

23. R. Gomory, *Outline of an algorithm for integer solutions to linear programs*, Bulletin of the American Mathematical Society, 64 (1958), pp. 275–278.

24. ———, *An algorithm for the mixed integer problem*, Tech. Rep. RM-2597, RAND Corporation, 1960.

25. ———, *An algorithm for integer solutions to linear programming*, in Recent Advances in Mathemtaical Programming, R. Graves and P. Wolfe, eds., McGraw-Hill, New York, 1963, pp. 269–302.

26. R. Jeroslow, *A cutting-plane game for facial disjunctive programs*, SIAM Journal on Control and Optimization, 18 (1980), pp. 264–281.

27. M. Jörg, *k-disjunctive cuts and a finite cutting plane algorithm for general mixed integer linear programs*, 2007. `http://arxiv.org/PS_cache/arxiv/pdf/0707/0707.3945v1.pdf`.

28. L. Lovász and A. Schrijver, *Cones of matrices and set-functions and 0-1 optimization*, SIAM Journal on Optimization, 1 (1991), pp. 166–190.

29. H. Marchand, A. Martin, R. Weismantel, and L. Wolsey, *Cutting planes in integer and mixed integer programming*, Discrete Applied Mathematics, 123 (2002), pp. 397–446.

30. G. Nemhauser and L. Wolsey, *Integer and combinatorial optimization*, John Wiley & Sons, New York, 1988.

31. ———, *A recursive procedure to generate all cuts for 0–1 mixed integer programs*, Mathematical Programming, 46 (1990), pp. 379–390.

32. J. Neto, *A simple finite cutting plane algorithm for integer programs*, Operations Research Letters, 40 (2012), pp. 578–580.

33. F. Nourie and E. Venta, *An upper bound on the number of cuts needed in Gomory's method of integer forms*, Operations Research Letters, 1 (1982), pp. 129–133.

34. L. Ntaimo, *Fenchel decomposition for stochastic mixed-integer programming*, Journal of Global Optimization, 55 (2013), pp. 141–163.

35. J. Orlin, *A finitely converging cutting plane technique*, Operations Research Letters, 4 (1985), pp. 1–3.

36. J. Owen and S. Mehrotra, *A disjunctive cutting plane procedure for general mixed-integer linear programs*, Mathematical programming, 89 (2001), pp. 437–448.

37. M. Padberg, *Classical cuts for mixed-integer programming and branch-and-cut*, Mathematical Methods of Operations Research, 53 (2001), pp. 173–203.

38. S. Sen and J. Higle, *The C^3 theorem and a D^2 algorithm for large scale stochastic mixed-integer programming: set convexification*, Mathematical Programming, 104 (2005), pp. 1–20.

39. S. Sen and H. Sherali, *On the convergence of cutting plane algorithms for a class of nonconvex mathematical programs*, Mathematical Programming, 31 (1985), pp. 42–56.

40. H. D. Sherali and W. P. Adams, *A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems*, SIAM Journal on Discrete Mathematics, 3 (1990), pp. 411–430.

41. ———, *A hierarchy of relaxations and convex hull representations for mixed-integer zero-one programming problems*, Discrete Applied Mathematics, 52 (1994), pp. 83–106.

42. H. D. Sherali and C. M. Shetty, *Optimization with Disjunctive Constraints*, Springer Verlag, Berlin, 1980.

43. W. White, *On Gomory's Mixed Integer Algorithm.* Senior Thesis, Princeton University, 1961.

44. A. Zanette, M. Fischetti, and E. Balas, *Lexicography and degeneracy: can a pure cutting plane algorithm work?*, Mathematical Programming, 130 (2010), pp. 1–24.